2022 IEEE International Conference on Robotics and Automation (ICRA)
May 23-27, 2022. Philadelphia, PA, USA

R^3LIVE: A Robust, Real-time, RGB-colored, LiDAR-Inertial-Visual tightly-coupled state Estimation and mapping package

Jiarong Lin and Fu Zhang

2022 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9681-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICRA46639.2022.9811935

***Abstract*— In this paper, we propose a novel LiDAR-Inertial-Visual sensor fusion framework termed R^3LIVE, which takes advantage of measurement of LiDAR, inertial, and visual sensors to achieve robust and accurate state estimation. R^3LIVE consists of two subsystems, a LiDAR-Inertial odometry (LIO) and a Visual-Inertial odometry (VIO). The LIO subsystem (FAST-LIO) utilizes the measurements from LiDAR and inertial sensors and builds the geometric structure (i.e., the positions of 3D points) of the map. The VIO subsystem uses the data of Visual-Inertial sensors and renders the map's texture (i.e., the color of 3D points). More specifically, the VIO subsystem fuses the visual data directly and effectively by minimizing the frame-to-map photometric error. The proposed system R^3LIVE is developed based on our previous work R^2LIVE, with a completely different VIO architecture design. The overall system is able to reconstruct the precise, dense, 3D, RGB-colored maps of the surrounding environment in real-time (see our attached video[1]). Our experiments show that the resultant system achieves higher robustness and accuracy in state estimation than its current counterparts. To share our findings and make contributions to the community, we open source R^3LIVE on our Github[2].**

I. Introduction

Recently, LiDAR sensors have been increasingly used in various robotic applications such as autonomous driving vehicles [1], drones [2, 3], etc. Especially with the emergence of low-cost solid-state LiDARs (e.g., [4]), more applications [5]–[9] based on these LiDARs propel the development of robotic community. However, for LiDAR-based SLAM systems, they would easily fail in those scenarios with not enough geometry features, especially for solid-state LiDARs which typically have a limited field of view (FoV) [10]. To address this problem, fusing LiDAR with other sensors such as camera [11]–[14] and Ultra-Wideband (UWB) [15, 16] can improve the system's robustness and accuracy. In particular, various of LiDAR-Visual fusion frameworks have been proposed recently in the robotics community [17]. V-LOAM [18] proposed by Zhang and Singh is one of the early works of LiDAR-Inertial-Visual systems, which leverage a loosely-coupled Visual-Inertial odometry (VIO) as the motion model for initializing the LiDAR mapping subsystem. Similarly, in [19], the authors propose a stereo Visual-Inertial LiDAR SLAM which incorporates the tightly-coupled stereo Visual-Inertial odometry with the LiDAR mapping and LiDAR enhanced visual loop closure. More recently, Wang *et al.* propose DV-LOAM [20], which is a direct Visual-LiDAR fusion framework. The system first utilizes a two-staged direct visual odometry module for efficient coarse state estimate, then the coarse pose is refined with the LiDAR mapping module, and finally the loop-closure module is leveraged for correcting the accumulated drift. The above systems fuse LiDAR-Inertial-Visual sensors at a level of loosely-coupled, in which the LiDAR measurement are not jointly optimized together with the visual or inertial measurements.

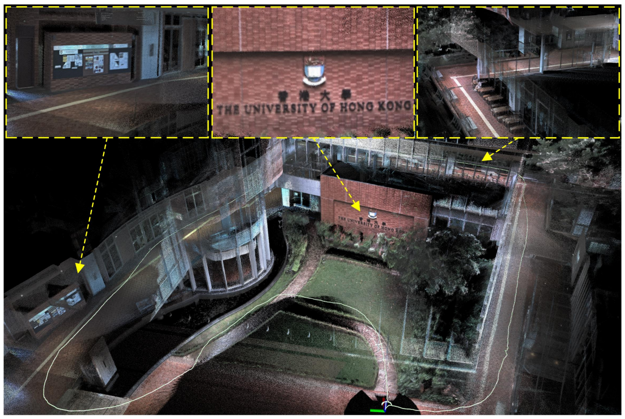

Fig. 1: Our developed R^3LIVE is able to reconstruct a dense, 3D, RGB-colored point cloud of the traveled environment in real-time. The white path is the figure is our traveling trajectory for collecting the data.

More tightly-coupled LiDAR-Inertial-Visual fusion frameworks have been proposed recently. For example, LIC-fusion [13] proposed by Zuo *et al.* is a tightly-coupled LiDAR-Inertial-Visual fusion framework, which combines the IMU measurements, sparse visual features, LiDAR features with online spatial and temporal calibration within the Multi-State Constrained Kalman Filter (MSCKF) framework. To further enhance the robustness of the LiDAR scan matching, their subsequent work termed LIC-Fusion 2.0 [14] proposes a plane-feature tracking algorithm across multiple LiDAR scans in a sliding-window and refines the pose trajectories within the window. Shan *et al.* in [12] propose LVI-SAM that fuses the LiDAR, visual and inertial sensors in a tightly-coupled smooth and mapping framework, which is built atop a factor graph. The LiDAR-Inertial and Visual-Inertial subsystems of LVI-SAM can function independently when failure is detected in one of them, or jointly when enough features are detected. Our previous work R^2LIVE [11] tightly fuses the data of LiDAR-Inertial-Visual sensors, extracts LiDAR and sparse visual features, estimates the state by minimizing the feature re-projection error within the framework of error-state iterated Kalman-filter for achieving the real-time performance, meanwhile improves the overall visual mapping accuracy with a sliding window optimization. R^2LIVE can run in various challenging scenarios with

[1] https://youtu.be/j5fT8NE5fdg

[2] https://github.com/hku-mars/r3live

aggressive motions, sensor failures, and even narrow tunnel-like environments with many moving objects with a small LiDAR FoV.

To further improve the real-time performance, robustness, and accuracy of our previous work R^{2}LIVE, we propose a novel sensor fusion framework termed R^{3}LIVE, which has a completely different VIO subsystem from R^{2}LIVE. In this paper, our contributions are:

- We propose a real-time simultaneous localization, mapping, and colorization framework. The presented framework consists of a LiDAR-Inertial odometry (LIO) for reconstructing geometry structure and a Visual-Inertial odometry (VIO) for texture rendering. The overall system is able to reconstruct a dense, 3D, RGB-colored point cloud of the environment in real-time (Fig. 1(a)).
- We propose a novel VIO system based on the RGB-colored point cloud map. The VIO estimates the current state by minimizing the photometric error between the RGB color of an observed map point and its measured color in the current image. Such a process requires no salient visual feature in the environment and saves the corresponding processing time (e.g., features detection and extraction), making our proposed system more robust, especially in texture-less environments.
- Thanks to the careful architecture designs and implementation, our proposed system achieves higher robustness and accuracy in state estimation than existing works. The overall system is validated in various indoor and outdoor environments. Results show that our system drifts only $0.16\,\mathrm{m}$ in translation and 3.9° in rotation after traveling up to $1.5\,\mathrm{km}$.
- To share our findings and make contributions to the community, we open source R^{3}LIVE on our Github[2], including all of our codes and the mechanical design of our handheld device.

II. The System Overview

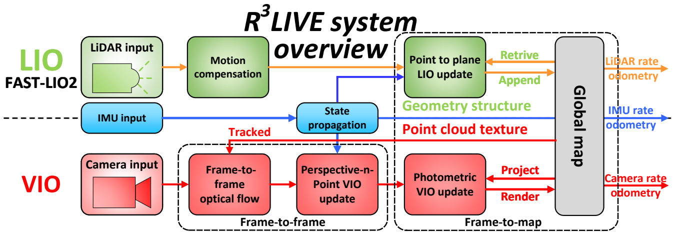

Fig. 2: The overview of our proposed system.

The overview of our system is shown in Fig. 2. Our proposed framework contains two subsystems: the LIO subsystem (upper part) and the VIO subsystem (the lower part). The LIO subsystem constructs the geometric structure of a global map, which registers the input LiDAR scan, and estimates the system's state by minimizing the point-to-plane residuals. The VIO subsystem builds the map's texture, which renders the RGB color of each point with the input image, updates the system state by minimizing the frame-to-frame PnP reprojection error and the frame-to-map photometric error.

TABLE I: NOMENCLATURE

Notation	Explanation
	Expression
$\boxplus/\boxminus$	The encapsulated "boxplus" and "boxminus" operations on manifold
$^G(\cdot)$	The value of $(\cdot)$ expressed in global frame
$^C(\cdot)$	The value of $(\cdot)$ expressed in camera frame
$\mathtt{Exp}(\cdot)/\mathtt{Log}(\cdot)$	The Rodrigues' transformation between the rotation matrix and rotation vector
$\delta(\cdot)$	The estimated error of $(\cdot)$. It is the minimum parameterization that characterized in the tangent space.
$\mathbf{\Sigma}_{(\cdot)}$	The covariance matrix of vector $(\cdot)$
	Variable
$\mathbf{b_g}, \mathbf{b_a}$	The bias of gyroscope and accelerometer
$^G\mathbf{g}$	The gravitational acceleration in global frame
$^G\mathbf{v}$	The linear velocity in global frame
$(^G\mathbf{R}_I, {^G\mathbf{p}_I})$	The attitude and position of the IMU w.r.t. global frame
$(^I\mathbf{R}_C, {^I\mathbf{p}_C})$	The extrinsic value between camera and IMU
$\mathbf{x}$	The full state vector
$\hat{\mathbf{x}}$	The prior estimation of $\mathbf{x}$
$\check{\mathbf{x}}$	The current estimate of $\mathbf{x}$ in each ESIKF iteration

III. Notation

In this paper, we use notations shown in TABLE I, which have been introduced in the previous work R^{2}LIVE [11].

A. State vector

In our work, we define the full state vector $\mathbf{x} \in \mathbb{R}^{29}$ as:

$$\mathbf{x} = \left[{^G\mathbf{R}_I^T}, {^G\mathbf{p}_I^T}, {^G\mathbf{v}^T}, \mathbf{b}_\mathbf{g}^T, \mathbf{b}_\mathbf{a}^T, {^G\mathbf{g}^T}, {^I\mathbf{R}_C^T}, {^I\mathbf{p}_C^T}, {^It_C}, \boldsymbol{\phi}^T\right]^T \quad (1)$$

where the notations $^G\mathbf{R}_I$, $^G\mathbf{p}_I$, $^G\mathbf{v}$, $\mathbf{b_g}$, $\mathbf{b_a}$, $^G\mathbf{g}$, $^I\mathbf{R}_C$, $^I\mathbf{p}_C$ are explained in Table I, ^It_C is the time-offset between IMU and camera while LiDAR is assumed to be synced with the IMU already, $\boldsymbol{\phi} = \left[f_x, f_y, c_x, c_y\right]^T$ are the camera intrinsics, where (f_x, f_y) denote the camera focal length and (c_x, c_y) the offsets of the principal point from the top-left corner of the image plane.

B. Maps representation

Our map is made up of voxels and points, where points are contained in voxels and are the minimum elements of maps.

1) Voxel: For fast finding the points in maps for rendering and tracking (see Section V-C and Section V-D) in our VIO subsystem, we design a fixed size (e.g. $0.1\,\mathrm{m}\times0.1\,\mathrm{m}\times0.1\,\mathrm{m}$) container named voxel. If a voxel has points appended recently (e.g. in recent one second), we mark this voxel as *activated*. Otherwise, this voxel is marked as *deactivated*.

2) Point: In our work, a point $\mathbf{P}$ is a vector with size 6:

$$\mathbf{P} = \left[{^G\mathbf{p}_x}, {^G\mathbf{p}_y}, {^G\mathbf{p}_z}, \mathbf{c}_r, \mathbf{c}_g, \mathbf{c}_b\right]^T = \left[{^G\mathbf{p}^T}, \mathbf{c}^T\right]^T \quad (2)$$

where the head 3×1 sub-vector $^G\mathbf{p} = \left[{^G\mathbf{p}_x}, {^G\mathbf{p}_y}, {^G\mathbf{p}_z}\right]^T$ denotes the point 3D position in the global frame, and the tail 3×1 vector $\mathbf{c} = \left[\mathbf{c}_r, \mathbf{c}_g, \mathbf{c}_b\right]^T$ is the point RGB color. Besides, we also record other informations of this point, such as the 3×3 covariance matrix $\mathbf{\Sigma_p}$ and $\mathbf{\Sigma_c}$, which denote the estimated errors of $^G\mathbf{p}$ and $\mathbf{c}$, respectively, and the timestamps when this point have been created and rendered.

IV. LiDAR-Inertial odometry subsystem

As shown in Fig. 2, the LIO subsystem of R^{3}LIVE constructs the geometric structure of the global map. For an incoming LiDAR scan, the motion distortion due to the continuous movement within the frame is compensated by IMU backward propagation, as shown in [5]. Then, we leverage the error-state iterated Kalman filter (ESIKF) that minimizes the point-to-plane residuals to estimate the system's state. Finally, with the converged state, the points of this scan are appended to the global map and mark the corresponding voxels as *activated* or *deactivated*. The accumulated 3D points in the global map form the geometry structure, which is also used for providing the depth for our VIO subsystem. For the detailed implementations of the LIO subsystem in R^{3}LIVE, we refer our readers to our previous related works [11, 21].

V. Visual-Inertial odometry subsystem

Our VIO subsystem renders the texture of a global map, estimating the system state by minimizing the photometric error. To be more specific, we project a certain number of points (i.e., tracked points) from the global map to the current image, then iteratively estimate the system state within the framework of ESIKF by minimizing the photometric error of these points. The tracked map points are sparse for the sake of efficiency. Minimizing the photometric error on these sparse map points usually requires building a pyramid of the input image. The pyramid is however not invariant to either translation or rotation, which is also needed to be estimated. In our proposed framework, we utilize the color of a single map point to compute the photometric error. The color, which is rendered simultaneously in the VIO, is an inherent property of the map point and is invariant to both translation and rotation of the camera. To ensure a robust and fast convergence, we design a two-step pipeline shown in Fig. 2. We firstly leverage a frame-to-frame optical flow to track map points and optimize the system's state by minimizing the Perspective-n-Point (PnP) projection error (Section V-A). Then, we further refine the system's state estimation by minimizing the frame-to-map photometric error among the tracked points (Section V-B). With the converged state estimate and the raw input image, we perform the texture rendering to update the colors of points in the global map (Section V-C).

A. Frame-to-frame Visual-Inertial odometry

Assume we have tracked m map points $\boldsymbol{\mathcal{P}} = \{\mathbf{P}_1, ..., \mathbf{P}_m\}$ in last image frame $\mathbf{I}_{k-1}$, and their projection on the $\mathbf{I}_{k-1}$ are $\{\boldsymbol{\rho}_{1_{k-1}}, ..., \boldsymbol{\rho}_{m_{k-1}}\}$, we leverage the Lucas−Kanade optical flow to find out their location in the current image frame $\mathbf{I}_k$, denoted as $\{\boldsymbol{\rho}_{1_k}, ..., \boldsymbol{\rho}_{m_k}\}$. Then, we calculate and optimize their projection error to estimate the state via an ESIKF.

1) Perspective-n-Point projection error: As shown in Fig. 3, we take the s-th point $\mathbf{P}_s = \left[{}^G\mathbf{p}_s^T, \mathbf{c}_s^T\right]^T \in \boldsymbol{\mathcal{P}}$ as example, the projection error $\mathbf{r}\left(\check{\mathbf{x}}_k, \boldsymbol{\rho}_s^k, {}^G\mathbf{p}_s\right)$ is calculated as follows:

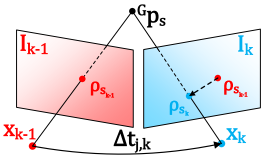

Fig. 3: Our frame-to-frame ESIKF VIO update the system's state with minimizing the PnP projection error.

$$
\begin{aligned}
{}^C\mathbf{p}_s = \left[{}^C\mathbf{p}_{x_s}, {}^C\mathbf{p}_{y_s}, {}^C\mathbf{p}_{z_s}\right]^T &= \left({}^G\check{\mathbf{R}}_{I_k} \cdot {}^I\check{\mathbf{R}}_{C_k}\right)^T \cdot {}^G\mathbf{p}_s \\
&\quad - {}^I\check{\mathbf{R}}_{C_k}^T \cdot {}^I\check{\mathbf{p}}_{C_k} - \left({}^G\check{\mathbf{R}}_{I_k} \cdot {}^I\check{\mathbf{R}}_{C_k}\right)^T \cdot {}^G\check{\mathbf{p}}_{I_k}
\end{aligned} \tag{3}
$$

$$
\mathbf{r}\left(\check{\mathbf{x}}_k, \boldsymbol{\rho}_{s_k}, {}^G\mathbf{p}_s\right) = \boldsymbol{\rho}_{s_k} - \boldsymbol{\pi}({}^C\mathbf{p}_s, \check{\mathbf{x}}_k) \tag{4}
$$

where $\check{\mathbf{x}}_k$ is the current state estimate of $\mathbf{x}$ in each ESIKF iteration, $\boldsymbol{\pi}({}^C\mathbf{p}_s, \check{\mathbf{x}}_k) \in \mathbb{R}^2$ is computed as below:

$$
\begin{aligned}
\boldsymbol{\pi}({}^C\mathbf{p}_s, \check{\mathbf{x}}_k) &= \left[\check{f}_{x_k}\frac{{}^C\mathbf{p}_{x_s}}{{}^C\mathbf{p}_{z_s}} + \check{c}_{x_k},\ \check{f}_{y_k}\frac{{}^C\mathbf{p}_{y_s}}{{}^C\mathbf{p}_{z_s}} + \check{c}_{y_k}\right]^T \\
&\quad + \frac{{}^I\check{t}_{C_k}}{\Delta t_{k-1,k}}(\boldsymbol{\rho}_{s_k} - \boldsymbol{\rho}_{s_{k-1}})
\end{aligned} \tag{5}
$$

with $\Delta t_{k-1,k}$ is the time interval between last image frame $\mathbf{I}_{k-1}$ and current image frame $\mathbf{I}_k$. Notice that in (5), the first item is the pin-hole projection function and the second one is the online-temporal correction factor [22].

The measurement noise in the residual (4) consists of two sources: one is the pixel tracking error in $\boldsymbol{\rho}_{s_k}$ and the other lies in the map point location error ${}^G\mathbf{p}_s$,

$$
{}^G\mathbf{p}_s = {}^G\mathbf{p}_s^{\mathtt{gt}} + \mathbf{n}_{\mathbf{p}_s},\ \mathbf{n}_{\mathbf{p}_s} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{n}_{\mathbf{p}_s}}) \tag{6}
$$

$$
\boldsymbol{\rho}_{s_k} = \boldsymbol{\rho}_{s_k}^{\mathtt{gt}} + \mathbf{n}_{\boldsymbol{\rho}_{s_k}},\ \mathbf{n}_{\boldsymbol{\rho}_{s_k}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{n}_{\boldsymbol{\rho}_{s_k}}}) \tag{7}
$$

where ${}^G\mathbf{p}_s^{\mathtt{gt}}$ and $\boldsymbol{\rho}_{s_k}^{\mathtt{gt}}$ are the true values of ${}^G\mathbf{p}_s$ and $\boldsymbol{\rho}_{s_k}$, respectively. Then, we obtain the first order Taylor expansion of the true zero residual $\mathbf{r}(\mathbf{x}_k, \boldsymbol{\rho}_{s_k}^{\mathtt{gt}}, {}^G\mathbf{p}_s^{\mathtt{gt}})$ as:

$$
\mathbf{0} = \mathbf{r}(\mathbf{x}_k, \boldsymbol{\rho}_{s_k}^{\mathtt{gt}}, {}^G\mathbf{p}_s^{\mathtt{gt}}) \approx \mathbf{r}\left(\check{\mathbf{x}}_k, \boldsymbol{\rho}_{s_k}, {}^G\mathbf{p}_s\right) + \mathbf{H}_s^r\delta\check{\mathbf{x}}_k + \boldsymbol{\alpha}_s \tag{8}
$$

where $\boldsymbol{\alpha}_s \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{\alpha}_s})$ and

$$
\mathbf{H}_s^r = \left.\frac{\partial\mathbf{r}_c(\check{\mathbf{x}}_k \boxplus \delta\check{\mathbf{x}}_k, \boldsymbol{\rho}_{s_k}, {}^G\mathbf{p}_s)}{\partial\delta\check{\mathbf{x}}_k}\right|_{\delta\check{\mathbf{x}}_k=\mathbf{0}} \tag{9}
$$

$$
\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_s} = \boldsymbol{\Sigma}_{\mathbf{n}_{\boldsymbol{\rho}_{s_k}}} + \mathbf{F}_{\mathbf{p}_s}^r\boldsymbol{\Sigma}_{\mathbf{p}_s}\mathbf{F}_{\mathbf{p}_s}^{r\ T}, \quad \mathbf{F}_{\mathbf{p}_s}^r = \frac{\partial\mathbf{r}(\check{\mathbf{x}}_k, \boldsymbol{\rho}_{s_k}, {}^G\mathbf{p}_s)}{\partial{}^G\mathbf{p}_s} \tag{10}
$$

2) Frame-to-frame VIO ESIKF update: Equation (8) constitutes an observation distribution for $\mathbf{x}_k$ (or equivalently $\delta\check{\mathbf{x}}_k \triangleq \mathbf{x}_k \boxminus \check{\mathbf{x}}_k$), which can be combined with the prior distribution from the IMU propagation to obtain the maximum a posteriori (MAP) estimation of $\delta\check{\mathbf{x}}_k$:

$$
\begin{aligned}
\min_{\delta\check{\mathbf{x}}_k} \Bigg(&\left\|\check{\mathbf{x}}_k \boxminus \hat{\mathbf{x}}_k + \boldsymbol{\mathcal{H}}\delta\check{\mathbf{x}}_k\right\|^2_{\boldsymbol{\Sigma}_{\delta\hat{\mathbf{x}}_k}} \\
&+ \sum\nolimits_{s=1}^{m}\left\|\mathbf{r}\left(\check{\mathbf{x}}_k, \boldsymbol{\rho}_{s_k}, {}^G\mathbf{p}_s\right) + \mathbf{H}_s^r\delta\check{\mathbf{x}}_k\right\|^2_{\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_s}} \Bigg)
\end{aligned} \tag{11}
$$

where $\|\mathbf{x}\|^2_{\boldsymbol{\Sigma}} = \mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}$ is the squared Mahalanobis distance with covariance $\boldsymbol{\Sigma}$, $\hat{\mathbf{x}}_k$ is the IMU propagated state estimate, $\boldsymbol{\Sigma}_{\delta\hat{\mathbf{x}}_k}$ is the IMU propagated state covariance, and $\boldsymbol{\mathcal{H}}$ is the Jacobian matrix when projecting the state error from the tangent space of $\hat{\mathbf{x}}_k$ to the tangent space of $\check{\mathbf{x}}_k$. The detailed derivation of first item in (11) can be found in Section E of R^{2}LIVE [11].

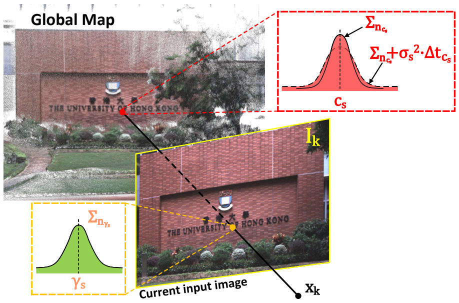
Global Map
$\Sigma_{\mathbf{n}_{\mathbf{c}_s}}$
$\Sigma_{\mathbf{n}_{\mathbf{c}_s}}+\sigma_s^2\cdot\Delta t_{\mathbf{c}_s}$
$\mathbf{c}_s$
$\mathbf{I}_k$
$\Sigma_{\mathbf{n}_{\gamma_s}}$
γ_s
Current input image
$\mathbf{x}_k$

Fig. 4: Our frame-to-map ESIKF VIO updates the system state by minimizing the photometric error between the map point color and its measured color in the current image.

Denote:

$$\mathbf{H} = \left[\mathbf{H}_1^{r\,T}, \ldots, \mathbf{H}_m^{r\;T}\right]^T, \quad \mathbf{R} = \mathrm{diag}(\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_1}, \ldots, \boldsymbol{\Sigma}_{\boldsymbol{\alpha}_m}) \tag{12}$$

$$\check{\mathbf{z}}_k = \left[\mathbf{r}\left(\check{\mathbf{x}}_k, \boldsymbol{\rho}_{1_k}, {}^G\mathbf{p}_1\right)\ldots, \mathbf{r}\left(\check{\mathbf{x}}_k, \boldsymbol{\rho}_{m_k}, {}^G\mathbf{p}_m\right)\right]^T \tag{13}$$

$$\mathbf{P} = (\boldsymbol{\mathcal{H}})^{-1}\boldsymbol{\Sigma}_{\delta\check{\mathbf{x}}_k}(\boldsymbol{\mathcal{H}})^{-T} \tag{14}$$

Following [5], we have the Kalman gain computed as:

$$\mathbf{K} = \left(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} + \mathbf{P}^{-1}\right)^{-1}\mathbf{H}^T\mathbf{R}^{-1} \tag{15}$$

Then we can update the state estimate as:

$$\check{\mathbf{x}}_k = \check{\mathbf{x}}_k \boxplus \left(-\mathbf{K}\check{\mathbf{z}}_k - (\mathbf{I}-\mathbf{KH})(\boldsymbol{\mathcal{H}})^{-1}(\check{\mathbf{x}}_k \boxminus \hat{\mathbf{x}}_k)\right) \tag{16}$$

The above process (Section V-A.1 to Section V-A.2) is iterated until convergence (i.e., the update is smaller than a given threshold). Notice that such an iterated Kalman Filter is known to be equivalent to a Gauss-Newton optimization [23].

B. Frame-to-map Visual-Inertial odometry

1) Frame-to-map photometric update: After the frame-to-frame VIO update, we have a good estimated state $\check{\mathbf{x}}_k$, then we perform the frame-to-map VIO update by minimizing the photometric error of the tracked points to lower the drift. As shown in Fig. 4, take the s-th tracked point $\mathbf{P}_s \in \boldsymbol{\mathcal{P}}$ as example, its frame-to-map photometric error $\mathbf{o}(\check{\mathbf{x}}_k, {}^G\mathbf{p}_s, \mathbf{c}_s)$ is calculated as follows:

$$\mathbf{o}(\check{\mathbf{x}}_k, {}^G\mathbf{p}_s, \mathbf{c}_s) = \mathbf{c}_s - \boldsymbol{\gamma}_s \tag{17}$$

where $\mathbf{c}_s$ is the point color saved in the global map, $\boldsymbol{\gamma}_s$ is the color observed in the current image $\mathbf{I}_k$. To obtain the observed color $\boldsymbol{\gamma}_s$ and its covariance $\boldsymbol{\Sigma}_{\mathbf{n}_{\boldsymbol{\gamma}_s}}$, we predict the point location $\tilde{\boldsymbol{\rho}}_{s_k} = \boldsymbol{\pi}({}^C\mathbf{p}_s, \check{\mathbf{x}}_k)$ on the current image $\mathbf{I}_k$ and then linearly interpolate the RGB colors of neighborhood pixels.

We also consider the measurement noise of $\boldsymbol{\gamma}_s$ and $\mathbf{c}_s$:

$$\boldsymbol{\gamma}_s = \boldsymbol{\gamma}_s^{gt} + \mathbf{n}_{\boldsymbol{\gamma}_s}, \;\; \mathbf{n}_{\boldsymbol{\gamma}_s} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{n}_{\boldsymbol{\gamma}_s}}) \tag{18}$$

$$\begin{aligned}\mathbf{c}_s = \mathbf{c}_s^{gt} + \mathbf{n}_{\mathbf{c}_s} + \boldsymbol{\eta}_{\mathbf{c}_s}, \;\; \mathbf{n}_{\mathbf{c}_s} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{n}_{\mathbf{c}_s}})\\ \boldsymbol{\eta}_{\mathbf{c}_s} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\sigma}_s^2\cdot\Delta t_{\mathbf{c}_s})\end{aligned} \tag{19}$$

where $\boldsymbol{\gamma}_s^{gt}$ is the ground true value of $\boldsymbol{\gamma}_s$ and $\mathbf{c}_s^{gt}$ is the true value of $\mathbf{c}_s$, $\Delta t_{\mathbf{c}_s}$ is the time interval between current frame and last rendering time of $\mathbf{P}_s$. Notice that in (19), the measurement noise of $\mathbf{c}_s$ consists of two parts: the estimated error $\mathbf{n}_{\mathbf{c}_s}$ from the last time of rendering (see Section V-C), and the impulsed random walk process noise $\boldsymbol{\eta}_{\mathbf{c}_s}$, which accounts for the change of environment illumination.

Combining (17), (18) and (19), we obtain the first order Taylor expansion of the true zero residual $\mathbf{o}(\mathbf{x}_k, {}^G\mathbf{p}_s^{gt}, \mathbf{c}_s^{gt})$:

$$\mathbf{0} = \mathbf{o}(\mathbf{x}_k, {}^G\mathbf{p}_s^{gt}, \mathbf{c}_s^{gt}) \approx \mathbf{o}(\check{\mathbf{x}}_k, {}^G\mathbf{p}_s, \mathbf{c}_s) + \mathbf{H}_s^o\delta\check{\mathbf{x}}_k + \boldsymbol{\beta}_s \tag{20}$$

where $\boldsymbol{\beta}_s \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}_s})$ and:

$$\mathbf{H}_s^o = \left.\frac{\partial\mathbf{o}(\check{\mathbf{x}}_k \boxplus \delta\check{\mathbf{x}}_k, {}^G\mathbf{p}_s, \mathbf{c}_s)}{\partial\delta\check{\mathbf{x}}_k}\right|_{\delta\check{\mathbf{x}}_k=\mathbf{0}} \tag{21}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{\beta}_s} = \boldsymbol{\Sigma}_{\mathbf{n}_{\mathbf{c}_s}} + \boldsymbol{\sigma}_s^2\cdot\Delta t_{\mathbf{c}_s} + \boldsymbol{\Sigma}_{\mathbf{n}_{\boldsymbol{\gamma}_s}} + \mathbf{F}_{\mathbf{p}_s}^o\boldsymbol{\Sigma}_{\mathbf{p}_s}\mathbf{F}_{\mathbf{p}_s}^{o\;T} \tag{22}$$

$$\mathbf{F}_{\mathbf{p}_s}^o = \frac{\partial\mathbf{o}(\check{\mathbf{x}}_k, {}^G\mathbf{p}_s, \mathbf{c}_s)}{\partial{}^G\mathbf{p}_s} \tag{23}$$

2) Frame-to-map VIO ESIKF update: Equation (20) constitutes another observation distribution for $\delta\check{\mathbf{x}}_k$, which is combined with the prior distribution from the IMU propagation to obtain the maximum a posteriori (MAP) estimation of $\delta\check{\mathbf{x}}_k$:

$$\begin{aligned}\min_{\delta\check{\mathbf{x}}_k}\Big(&\|\check{\mathbf{x}}_k \boxminus \hat{\mathbf{x}}_k + \boldsymbol{\mathcal{H}}\delta\check{\mathbf{x}}_k\|^2_{\boldsymbol{\Sigma}_{\delta\check{\mathbf{x}}_k}}\\ &+\sum\nolimits_{s=1}^{m}\left\|\mathbf{o}(\check{\mathbf{x}}_k, {}^G\mathbf{p}_s, \mathbf{c}_s) + \mathbf{H}_s^o\delta\check{\mathbf{x}}_k\right\|^2_{\boldsymbol{\Sigma}_{\boldsymbol{\beta}_s}}\Big)\end{aligned} \tag{24}$$

Denote $\mathbf{H}, \mathbf{R}, \check{\mathbf{z}}_k$ and $\mathbf{P}$ similar to (12) $\sim$ (14) :

$$\mathbf{H} = \left[\mathbf{H}_1^{oT}, \ldots, \mathbf{H}_m^{o\;T}\right]^T, \quad \mathbf{R} = \mathrm{diag}(\boldsymbol{\Sigma}_{\boldsymbol{\beta}_1}, \ldots, \boldsymbol{\Sigma}_{\boldsymbol{\beta}_m}) \tag{25}$$

$$\check{\mathbf{z}}_k = \left[\mathbf{o}(\check{\mathbf{x}}_k, {}^G\mathbf{p}_1, \mathbf{c}_1)\ldots, \mathbf{o}(\check{\mathbf{x}}_k, {}^G\mathbf{p}_m, \mathbf{c}_m)\right]^T \tag{26}$$

$$\mathbf{P} = (\boldsymbol{\mathcal{H}})^{-1}\boldsymbol{\Sigma}_{\delta\check{\mathbf{x}}_k}(\boldsymbol{\mathcal{H}})^{-T} \tag{27}$$

Then we perform the state update similar to (15) and (16). This frame-to-map VIO ESIKF update (Section V-B.1 to Section V-B.2) is iterated until convergence. The converged state estimate is then used to: (1) render the texture of maps (Section V-C); (2) update the current tracking point set $\boldsymbol{\mathcal{P}}$ for the next frame to use (Section V-D); and (3) serve as the starting point of the IMU propagation in the next frame of LIO or VIO update.

$$\hat{\mathbf{x}}_k = \check{\mathbf{x}}_k, \quad \boldsymbol{\Sigma}_{\delta\hat{\mathbf{x}}_k} = (\mathbf{I}-\mathbf{KH})\,\boldsymbol{\Sigma}_{\delta\check{\mathbf{x}}_k} \tag{28}$$

C. Render the texture of global map

After the frame-to-map VIO update, we have the precise pose of the current image, and then we perform the rendering function to update the color of map points.

First of all, we retrieve all the points in all *activated* voxels (see Section IV). Assume the total number points is n points and the point set $\boldsymbol{\zeta} = \{\mathbf{P}_1, \ldots, \mathbf{P}_n\}$. Taking the color update process of the s-th point $\mathbf{P}_s = \left[{}^G\mathbf{p}_s^T, \mathbf{c}_s^T\right]^T \in \boldsymbol{\zeta}$ as an example. If $\mathbf{P}_s$ falls in the current image $\mathbf{I}_k$, we obtain its observation color $\boldsymbol{\gamma}_s$ and covariance $\boldsymbol{\Sigma}_{\mathbf{n}_{\boldsymbol{\gamma}_s}}$ by linearly interpolating the RGB values of the neighborhood pixels on $\mathbf{I}_k$. The newly observed point color is fused with the existing color $\mathbf{c}_s$ recorded in the map via Bayesian update, leading to the updated color and the associated covariance of $\mathbf{c}_s$ as follows:

$$\boldsymbol{\Sigma}_{\mathbf{n}_{\tilde{\mathbf{c}}_s}} = \left(\left(\boldsymbol{\Sigma}_{\mathbf{n}_{\mathbf{c}_s}} + \boldsymbol{\sigma}_s^2\cdot\Delta t_{\mathbf{c}_s}\right)^{-1} + \boldsymbol{\Sigma}_{\mathbf{n}_{\boldsymbol{\gamma}_s}}^{-1}\right)^{-1} \tag{29}$$

$$\tilde{\mathbf{c}}_s = \left(\left(\boldsymbol{\Sigma}_{\mathbf{n}_{\mathbf{c}_s}} + \boldsymbol{\sigma}_s^2\cdot\Delta t_{\mathbf{c}_s}\right)^{-1}\mathbf{c}_s + \boldsymbol{\Sigma}_{\mathbf{n}_{\boldsymbol{\gamma}_s}}^{-1}\boldsymbol{\gamma}_s\right)^{-1}\boldsymbol{\Sigma}_{\mathbf{n}_{\tilde{\mathbf{c}}_s}} \tag{30}$$

$$\mathbf{c}_s = \tilde{\mathbf{c}}_s, \quad \boldsymbol{\Sigma}_{\mathbf{n}_{\mathbf{c}_s}} = \boldsymbol{\Sigma}_{\mathbf{n}_{\tilde{\mathbf{c}}_s}} \tag{31}$$

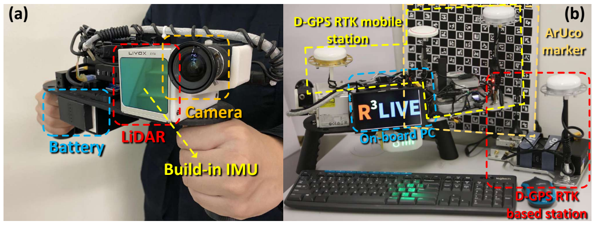

Fig. 5: Our handheld device for data collection. (a) shows our minimum system. (b) shows an additional D-GPS RTK system and an ArUco marker board used to evaluate the system accuracy.

D. Update of the tracking points of VIO subsystem

After the rendering of texture, we perform the update of tracked point set $\mathcal{P}$. Firstly, we remove the points from $\mathcal{P}$ if their projection error in (4) or photometric error in (17) are too large, and also remove the points which does not fall into $\mathbf{I}_k$. Secondly, we project each point in $\boldsymbol{\zeta}$ to the current image $\mathbf{I}_k$, and add it to $\mathcal{P}$ if there have no other tracked points located nearby (e.g. in the radius of 50 pixels).

VI. Experiments and Results

A. Setup of the Experiments

1) Handheld device for data collection: Our handheld device for data collection is shown in Fig. 5(a), which includes a power supply unit, the onboard DJI *manifold-2c* computation platform (equipped with an *Intel i7-8550u* CPU and 8 GB RAM), a *FLIR Blackfly BFS-u3-13y3c* global shutter camera, and a *LiVOX AVIA* LiDAR. The FoV of the camera is $82.9\,^{\circ}\times 66.5^{\circ}$ while the FoV of the LiDAR is $70.4\,^{\circ}\times 77.2^{\circ}$. To quantitatively evaluate the precision of our algorithm (Section VI-D), we install a Differential-GPS (D-GPS) real-time kinematic (RTK) system [24] on the device, shown in Fig. 5(b). In GPS denied environment (Section VI-B and Section VI-C), we use the ArUco marker [25] for calculating the drift of our odometry. Notice that all of the mechanical modules of this device are designed as FDM printable, and to facilitate readers to reproduce the work, we open source all of the schematics and the collected datasets on Github[2].

2) Configuration of parameters: All experiments in this paper are conducted with the same parameter settings (see the configuration file at [26]). In the initialization of $\mathbf{x}$ in (1), we set the intrinsic and extrinsic values (i.e., $\boldsymbol{\phi}$, ${}^{I}\mathbf{R}_{C}^{T}$, ${}^{I}\mathbf{p}_{C}^{T}$) to the offline calibration results, and the rest state components are initialized as zeros (for pose, velocity and bias) or the first accelerometer readings (for gravity).

B. Experiment-1: Robustness evaluation in simultaneously LiDAR degenerated and visual texture-less environments

In this experiment, we challenge one of the most difficult scenarios in SLAM. We perform the robustness evaluation in simultaneously LiDAR degenerated and visual texture-less environments. As shown in Fig. 6, our sensors pass through a narrow "T"-shape passage while occasionally facing against the side walls. When facing a wall, which imposes only a single plane constraint, the LiDAR is well-known degenerated in estimating the full pose. Moreover, the visual texture on the white walls is very limited (Fig. 6(a) and Fig. 6(c)), especially for the wall-1, which has the only changes of illumination. The absence of available LiDAR and visual features lead such scenarios to be rather challenging for both LiDAR-based and visual-based SLAM methods.

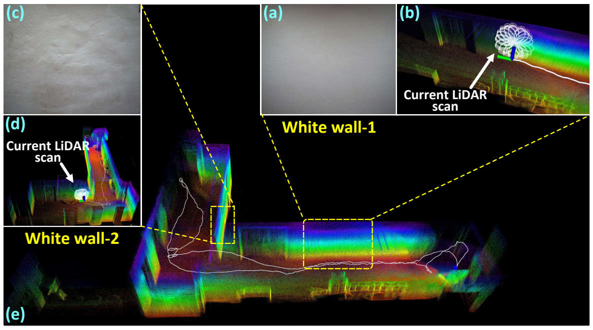

Fig. 6: We test our algorithm in simultaneously LiDAR degenerated and visual texture-less scenarios.

By taking advantage of using the raw image pixel color information, our proposed algorithm can "survive" in such extremely difficult scenarios. Fig. 8 shows our estimated pose, with the phases of passing through "wall-1" and "wall-2" shaded with purple and yellow, respectively. The estimated covariance is also shown in Fig. 8, which are bounded over the whole estimated trajectory, indicating that our estimation quality is stable over the whole process. The sensors are moved to the starting point, where an ArUco marker board is used to obtain the ground-truth relative pose between the starting and end poses. Compared with the ground-truth end pose, our algorithm drifts 1.62° in rotation and $4.57\,\mathrm{cm}$ in translation. We recommend the readers to the accompanying video on YouTube[1] for better visualization of the experiment.

C. Experiment-2: High precision mapping large-scale indoor & outdoor urban environment

In this experiment, we show the capacity of our algorithm for reconstructing a precise, dense, 3D, RGB-colored map of a large-scale campus environment in real-time. We collect the data within the campus of the Hong Kong University of Science and Technology (HKUST) 4 times with different traveling trajectories (i.e., Traj $1\sim4$), with their total length are $1191\,\mathrm{m}$, $1372\,\mathrm{m}$, $1524\,\mathrm{m}$ and $1191\,\mathrm{m}$, respectively. The bird-view (i.e., project on $\mathrm{X-Y}$ plane) of these trajectories are shown Fig. 7(f). Without any additional processing (e.g., loop closure), all of these four trajectories can close the loop itself (see Fig. 7(e)). Using the ArUco marker board placed at the starting point, the odometry drifts are presented in Table II, which indicates that our proposed method is of high accuracy with small drifts over the long trajectories and in complicated environments. Finally, we present the the reconstructed map in "Traj-1" in Fig. 7. More visualization results are available on the project page [27].

TABLE II: Odometry drift in 4 trajectories of Experiment-2.

	Traj-1	Traj-2	Traj-3	Traj-4
Length of trajectory (m)	1317	1372	1524	1191
Drift in translation (m)	0.093	0.164	0.154	0.102
Drift in rotation (deg)	2.140	2.342	0.285	3.925

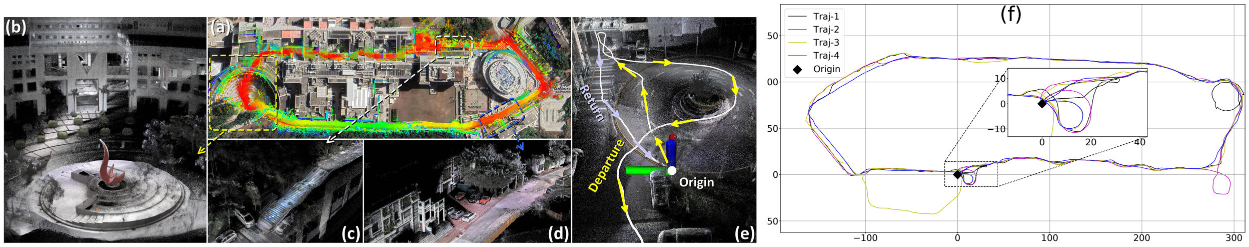

Fig. 7: We show the dense 3D RGB-colored maps reconstructed in real-time in Experiment-2. In (a), we merge our maps with the Google-Earth satellite image and find them aligned well. Details of the map are selectively shown in (b), (c), and (d), respectively. In (e), we show that our algorithm can close the loop itself without any additional processing (e.g., loop-detection and closure), with the return trajectory can return back to the origin. We refer readers to the video on our YouTube[1]. (f) plot the birdview of 4 traveling trajectories.

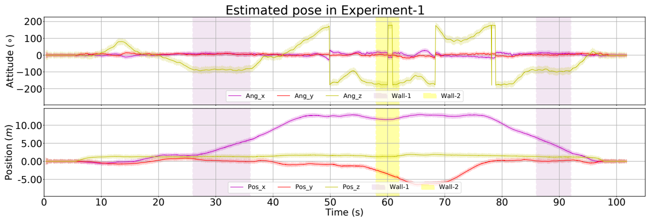

Fig. 8: The estimated poses and their 3-σ bound with 5 times amplification for better visualization (the light-colored area around the trajectory) of Experiment-1. The shaded areas in purple and yellow are the phases of the sensors facing against the white "wall-1" and "wall-2", respectively.

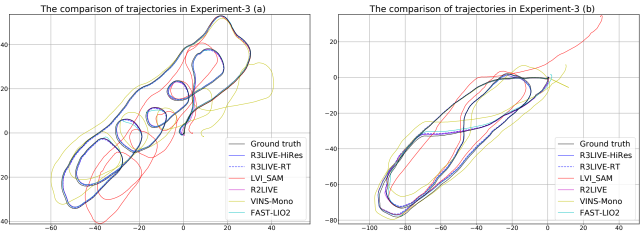

Fig. 9: Comparison of the estimated trajectories in Experiments-3.

D. Experiment-3: Quantitative evaluation of precision using D-GPS RTK

In this experiment, we collect two sequences of data in a seaport (*Belcher Bay Promenade* [28]) with a real-time Differential-GPS (D-GPS) Kinematic (RTK) system providing the ground-truth trajectory. In these two sequences, the sensors often face many pedestrians and occasionally open seas, where the LiDAR has very few measurements. We compare the trajectories estimated by R^{3}LIVE with two different configurations ("R3LIVE-HiRES" and "R3LIVE-RT", see Table III), "LVI-SAM" [12] (with modified its LiDAR front-end for Livox Avia LiDAR), "R2LIVE" [11], "VINS-Mono" [29], "Fast-LIO2" [21] with the ground-truth in Fig. 9, where we can see that our estimated trajectories agree the best with the ground-truth in both sequences. To have more quantitative comparisons, we compute the relative rotation error (RPE) and relative translation error (RTE) [30] over all possible sub-sequences of length (50,100,150,...,300) meters are shown in Table III.

E. Run time analysis

We investigate the average time consumption among our system all experiments on two different platforms: the desktop PC (with Intel i7-9700K CPU and 32GB RAM) and a UAV on-board computer ("OB", with Intel i7-8550u CPU and 8GB RAM). The detailed statistics are listed in Table IV. The time consumption of our VIO subsystem is affected by two main settings: the size of an image ("Image size") and the resolution of point cloud map ("Pc res").

TABLE III: The relative pose error in Experiment-3. In configuration "R3LIVE-HiRES", we set the input image resolution as 1280×1024 and the minimum distance (i.e., the resolution of point cloud map) between two points in maps as $0.01\,\mathrm{m}$. In configuration "R3LIVE-RT", we set the input image resolution as 320×256 and the minimum distance between two points in maps as $0.10\,\mathrm{m}$

Relative pose error in Experiments-3 (a)						
Sub-sequence RRE/RTE	50 m deg / %	100 m deg / %	150 m deg / %	200 m deg / %	250 m deg / %	300 m deg / %
R3LIVE-HiRES	**0.99** / 2.25	**0.53** / **1.02**	**0.46** / **0.60**	**0.29** / **0.31**	**0.24** / 0.31	**0.21** / **0.17**
R3LIVE-RT	1.48 / **2.21**	0.54 / 1.06	0.49 / 0.60	0.31 / 0.41	0.25/ **0.31**	0.22 / 0.23
LVI_SAM	2.11 / 13.73	1.04 / 6.69	0.83 / 5.07	0.60 / 3.86	0.47 / 2.98	0.43 / 2.40
R2LIVE	1.21 / 2.47	0.61 / 1.02	0.59 / 0.60	0.34 / 0.34	0.37 / 0.33	0.34 / 0.21
FAST-LIO2	1.36 / 2.35	0.66 / 0.82	0.47 / 0.60	0.37 / 0.36	0.37 / 0.31	0.21 / 0.20
VINS-Mono	3.03 / 10.41	1.70 / 7.63	1.15 / 6.36	0.89 / 4.34	0.73 / 3.08	0.59 / 2.31
Relative pose error in Experiments-3 (b)						
Sub-sequence RRE/RTE	50 m deg / %	100 m deg / %	150 m deg / %	200 m deg / %	250 m deg / %	300 m deg / %
R3LIVE-HiRES	1.06 / **1.98**	0.58 / **1.34**	**0.43** / **0.83**	**0.32** / **0.59**	0.27 / **0.27**	0.25 / 0.16
R3LIVE-RT	**0.98** / 2.08	**0.55** / 1.61	0.47 / 0.99	0.39 / 0.65	0.33 / 0.33	0.25 / **0.14**
LVI_SAM	2.04 / 3.33	1.05 / 2.37	0.81 / 1.53	0.57 / 1.38	0.49 / 1.13	0.44 / 1.01
R2LIVE	1.13 / 2.06	0.65 / 1.45	0.49 / 0.88	0.31 / 0.57	0.26 / 0.30	0.29 / 0.16
FAST-LIO2	1.31 / 2.22	0.69 / 1.34	0.49 / 0.89	0.31 / 0.59	**0.26** / 0.31	**0.25** / 0.29
VINS-Mono	2.44 / 9.35	1.31 / 7.28	0.99 / 4.63	0.66 / 3.68	0.56 / 3.24	0.48 / 2.73

TABLE IV: The average time consumption on two different platforms with different configurations.

VIO per-frame cost time										LIO per-frame cost time
Image size	320×256			640×512			1280×1024			
Pc res (m)	0.10	0.05	0.01	0.10	0.05	0.01	0.10	0.05	0.01	
On PC (ms)	7.08	8.43	16.78	10.38	11.74	20.67	12.13	13.19	23.17	18.40
On OB (ms)	15.30	21.69	43.79	23.71	34.70	47.99	30.38	42.43	61.34	39.56

VII. Conclusions and Future Work

In this paper, we proposed a novel LiDAR-Inertial-Visual sensor fusion framework named R^{3}LIVE that consisted of two subsystems: a LIO and a VIO subsystem. The LIO subsystem constructs the geometric structure of global map while the VIO subsystem renders the map's texture. We validate the robustness of the proposed system in a simultaneously LiDAR degenerated and visual texture-less environment, and evaluate the accuracy with various large-scale, indoor, and outdoor environments. The experiment results proved that R^{3}LIVE could achieve higher robustness and accuracy than existing state-of-the-art SLAM systems. In the future, we will further improve the performance and robustness of R^{3}LIVE by adding online photometric calibration.

References

[1] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *2011 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2011, pp. 163–168.

[2] F. Gao, W. Wu, W. Gao, and S. Shen, "Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments," *Journal of Field Robotics*, vol. 36, no. 4, pp. 710–733, 2019.

[3] F. Kong, W. Xu, Y. Cai, and F. Zhang, "Avoiding dynamic small obstacles with onboard sensing and computation on aerial robots," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7869–7876, 2021.

[4] Z. Liu, F. Zhang, and X. Hong, "Low-cost retina-like robotic lidars based on incommensurable scanning," *IEEE Transactions on Mechatronics*, 2021, in press.

[5] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robotics and Automation Letters*, 2021, in press.

[6] J. Lin, X. Liu, and F. Zhang, "A decentralized framework for simultaneous calibration, localization and mapping with multiple lidars," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4870–4877.

[7] Z. Liu and F. Zhang, "Balm: Bundle adjustment for lidar mapping," *IEEE Robotics and Automation Letters*, 2021, in press.

[8] X. Liu and F. Zhang, "Extrinsic calibration of multiple lidars of small fov in targetless environments," *IEEE Robotics and Automation Letters*, 2021, in press.

[9] C. Yuan, X. Liu, X. Hong, and F. Zhang, "Pixel-level extrinsic self calibration of high resolution lidar and camera in targetless environments," *arXiv preprint arXiv:2103.01627*, 2021.

[10] J. Lin and F. Zhang, "Loam livox: A fast, robust, high-precision lidar odometry and mapping package for lidars of small fov," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3126–3131.

[11] J. Lin, C. Zheng, W. Xu, and F. Zhang, "R^{2}live: A robust, real-time, lidar-inertial-visual tightly-coupled state estimator and mapping," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7469–7476, 2021.

[12] T. Shan, B. Englot, C. Ratti, and D. Rus, "Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping," *arXiv preprint arXiv:2104.10831*, 2021.

[13] X. Zuo, P. Geneva, W. Lee, Y. Liu, and G. Huang, "Lic-fusion: Lidar-inertial-camera odometry," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 5848–5854.

[14] X. Zuo, Y. Yang, J. Lv, Y. Liu, G. Huang, and M. Pollefeys, "Lic-fusion 2.0: Lidar-inertial-camera odometry with sliding-window plane-feature tracking," in *IROS 2020*, 2020.

[15] W. Zhen and S. Scherer, "Estimating the localizability in tunnel-like environments using lidar and uwb," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4903–4908.

[16] H. Zhou, Z. Yao, and M. Lu, "Uwb/lidar coordinate matching method with anti-degeneration capability," *IEEE Sensors Journal*, vol. 21, no. 3, pp. 3344–3352, 2020.

[17] C. Debeunne and D. Vivet, "A review of visual-lidar fusion based simultaneous localization and mapping," *Sensors*, vol. 20, no. 7, p. 2068, 2020.

[18] J. Zhang and S. Singh, "Laser–visual–inertial odometry and mapping with high robustness and low drift," *Journal of Field Robotics*, vol. 35, no. 8, pp. 1242–1264, 2018.

[19] W. Shao, S. Vijayarangan, C. Li, and G. Kantor, "Stereo visual inertial lidar simultaneous localization and mapping," *arXiv preprint arXiv:1902.10741*, 2019.

[20] W. Wang, J. Liu, C. Wang, B. Luo, and C. Zhang, "Dv-loam: Direct visual lidar odometry and mapping," *Remote Sensing*, vol. 13, no. 16, p. 3340, 2021.

[21] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *arXiv preprint arXiv:2107.06829*, 2021.

[22] T. Qin and S. Shen, "Online temporal calibration for monocular visual-inertial systems," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3662–3669.

[23] B. M. Bell and F. W. Cathey, "The iterated kalman filter update as a gauss-newton method," *IEEE Transactions on Automatic Control*, vol. 38, no. 2, pp. 294–297, 1993.

[24] A differential-GPS real-time kinematic (RTK) system: https://www.dji.com/d-rtk.

[25] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[26] Configuration: https://github.com/hku-mars/r3live/blob/master/config/r3live_config.yaml.

[27] Github: https://github.com/ziv-lin/r3live_preview.

[28] Google Map: https://goo.gl/maps/eikg6Kvg9k4gM3Sm9.

[29] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[30] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 7244–7251.